REMARKS

This is intended as a full and complete response to the Final Office Action dated December 18, 2008, having a shortened statutory period for response set to expire on March 18, 2009. Applicants submit this response to place the application in condition for allowance or in better form for appeal. Please reconsider the claims pending in the application for reasons discussed below.

Claims 1-4 and 7-21 are pending in the application. Claims 1-4 and 7-21 remain pending following entry of this response.

Claim Rejections - 35 U.S.C. § 103

Claims 1-4, 7-9, 11-17, and 19-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over *Cox et al.* (U.S. Publication No. 20020156806, hereinafter, "*Cox*") in view of *Baudel* (U.S. Patent No. 6,928,436).

Claims 10 and 18 are rejected under 35 U.S.C. 103(a) as being unpatentable over *Cox* in view of *Baudel*, further in view of *Koselj et al.* (U.S. Patent No. 7,027,056, hereinafter, "*Koselj*").

Applicants respectfully traverse this rejection.

The Examiner bears the initial burden of establishing a prima facie case of obviousness. See MPEP § 2141. Establishing a prima facie case of obviousness begins with first resolving the factual inquiries of Graham v. John Deere Co. 383 U.S. 1 (1966). The factual inquiries are as follows:

(A) determining the scope and content of the prior art;

(B) ascertaining the differences between the claimed invention and the prior art;

(C) resolving the level of ordinary skill in the art; and

(D) considering any objective indicia of nonobviousness.

Once the Graham factual inquiries are resolved, the Examiner must determine whether the claimed invention would have been obvious to one of ordinary skill in the art.

Respectfully, Applicants submit that the Examiner has not properly characterized the teachings of the references and/or the claims at issue. Accordingly, a prima facie case of obviousness has not been established.

For example, the Examiner continues to suggest that *Cox* discloses transformation rules for transforming the abstract data structure into a concrete data structure . . . the transformation rules describing graphical attributes of a requested graphical representation type as recited in claim 1. Specifically, the Examiner asserts as follows:

> Cox shows a computer implemented method of generating a graphical representation of data, comprising . . . retrieving and providing transformation rules for transforming the abstract data structure into a concrete data structure, the transformation rules comprising a plurality of subsets of transformation rules each subset describing graphical attributes of a requested graphical representation type (page 5, paragraph [0044], lines 12-23) . . . .

Final Office Action dated December 18, 2008, pages 2-3; *see also* Final Office Action dated September 25, 2007, page 3. The Examiner analogizes "the raw data being analyzed as the abstract data structure, and the implementation of the visualization… as the concrete data structure." *See, e.g.*, Final Office Action dated September 25, 2007, page 3. The Examiner also analogizes the "actions" in *Cox* to "transformation rules." *See id.* The "actions" are described in *Cox* as being selectable command options, presented to a user in "a list or through a graphical user interface," which "allow a user to change view parameters, select a subset of the author's data for viewing, or select specific data for display". *See Cox*, ¶ 44.

However, Applicants respectfully submit that the Examiner's analogy fails to explain how the cited material, as well as *Cox* generally, discloses the above limitation. For example, the Examiner fails to explain how a list of user command options (i.e., "actions") could represent the recited *subsets of transformation rules. See* Advisory Action dated December 7, 2007 ("each action is a subset of a plurality of transformation rules (e.g., a view parameter may be changed, so that data displayed as a bar graph may be displayed as a pie chart) . . . ."). Applicants respectfully submit that a user command option to "select specific data for display" or to "change a view parameter" is not analogous to a transformation rule. Thus, the recited limitations are not disclosed by *Cox*. Accordingly, Applicants submit that the rejection is defective and should be withdrawn.

Further, even assuming, *arguendo*, that a listing of user "actions" is somehow analogous to the limitation of *subsets of transformation rules*, this analogy does not conform to the remaining limitations of the present claims. For example, *Cox* does not teach that the abstract data structure (read as "raw data") in any way defines *a plurality of abstract attributes representing an abstract graphical representation of the data*. Moreover, *Cox* does not teach that each of the listed actions (read as "subset of transformation rules") in any way *describes graphical attributes of a requested graphical representation type*. The Examiner continues to suggest the contrary. *See, e.g.*, Advisory Action dated December 7, 2007 ("each action describes a graphical attribute of a requested graphical representation type (e.g., again, a view parameter may be changed, so that data displayed as a bar graph may be displayed as a pie chart) . . . ."). However, Applicants respectfully submit that a user command option to "change a view parameter" in no way *describes graphical attributes of a requested graphical representation type*. Thus, the recited limitations are not disclosed by *Cox*. Accordingly, Applicants submit that the rejection is defective and should be withdrawn.

Further, the Examiner continues to suggest that *Cox* discloses the limitation of abstract data structure templates, each . . . associated with a specific graphical representation type as recited in claim 2. Specifically, the Examiner asserts as follows:

> Cox shows a computer implemented method of generating a graphical representation of data, comprising . . . providing and selecting an abstract data template (e.g., Bar Chart view object) from a plurality of abstract data structure templates (e.g., dynamic tables, text objects) (abstract) . . . .

Final Office Action dated December 18, 2008, pages 2-3; *see also* Final Office Action dated September 25, 2007, page 3. That is, the Examiner continues to suggest that the recited *abstract data structure templates* are disclosed by, *e.g.*, a "BarChart view object." *Id.* Significantly, the Examiner's analogy fails to conform to the other limitations of claim 2. That is, claim 2 requires a limitation of *the abstract data structure being generated using the selected abstract data structure template.* The Examiner analogizes the "raw data being analyzed as the abstract data structure." *See* Final Office Action dated September 25, 2007, page 7. Respectfully, the Examiner's analogy leads to a contradictory result and is therefore untenable. That is, the Examiner's

analogy requires the raw data ("abstract data structure") to be generated using a view object such as "BarChart" ("abstract data structure templates"). Such a requirement is wholly contradictory and is not disclosed by (or even consistent with) *Cox*. That is, *Cox* does not describe "raw data" as being generated with the "view objects." Therefore, contrary to the Examiner's suggestion, *Cox* does not disclose *abstract data structure templates, each . . . associated with a specific graphical representation type*. Accordingly, Applicants submit that the rejection is defective and should be withdrawn.

Further, the Examiner suggests that Cox discloses generating, on the basis of the abstract data structure . . . a concrete data structure defining a concrete graphical representation of the data in a graphics rendering language using the transformation rules as recited in claim 1. Specifically, the Examiner states:

> Cox shows a computer implemented method of generating a graphical representation of data, comprising . . . generating, on the basis of the abstract data structure and the selected subset of transformation, a concrete data structure defining a concrete graphical representation in a graphics rendering language using the transformation rules; wherein generating the concrete data structure is done by operation of a computer processor (figure 8, and corresponding description) . . . .

Final Office Action dated December 18, 2008, page 3; *see also* Final Office Action dated September 25, 2007, page 3. That is, the Examiner analogizes a "live document display output" of *Cox*, Figure 8 to teach *generating . . . a concrete data structure* . . . . In other words, the Examiner's interpretation merely equates a "concrete data structure" with an "display output." Respectfully, the Examiner's interpretation trivializes the limitation of "generating . . . a concrete data structure defining a concrete graphical representation of the data in a graphics rendering language" as recited in claim 1. Although a display output may be a *result* of rendering according to a graphics rendering language, a display output does not define anything in any graphics rendering language.

Further, the description corresponding to Figure 8 in *Cox* (¶ 80-97) merely shows HTML code (i) for defining controls (¶ 83-87); and (ii) for invoking an applet (¶ 88-97). According to *Cox*:

> These controls are located within the text explaining their usage and in close proximity to the view bar chart view 832.
>
> Readers can also easily view the drivers with the most prize money by interacting with the "total winnings" histogram of view 834 or the related JavaScript controls . . . .
>
> . . . .
>
> ```
> <applet name=wins code=Idoc.BarApplet.class width=175
> height=130>
> <param name=url value="drivers.txt">
> <param name="Variable" value="Wins">
> </applet>
> ```

Cox, ¶ 80-81 and 94-97. As the cited passage illustrates, the controls in *Cox* are separate from the views in *Cox*, and are merely used to adjust the views. Therefore, following the Examiner's analogy (regarding the concrete data structure) yields a contradictory result, because the HTML code is not for generating a view (of the concrete data structure), but for defining a control, which is different from the view.

Further, the HTML code for invoking an applet merely invokes a Java applet (with two parameters: "url" and "Variable"). A Java applet is a software component that can run in a web browser. Moreover, the parameter "url value='drivers.txt'" is a reference to a text file containing raw data. In other words, the HTML code for invoking the applet is simply not any graphical representation of data. That is, the HTML code for invoking the applet does not describe graphics at all. Merely *invoking a software component and supplying the software component with a filename of a file containing raw data* is not the same as a concrete graphical representation in a graphics rendering language. Further, even assuming, *arguendo*, that *Cox* somehow discloses *a concrete graphical representation in a graphics rendering language*, Cox nevertheless fails to disclose that the concrete graphical representation is generated using the transformation rules, as required by claim 1. Therefore, for the reasons set forth above, individually and collectively, *Cox* does not disclose *generating, on the basis of the abstract data structure . . . a concrete data structure defining a concrete graphical representation of the data in a graphics rendering language using the transformation rules*. Accordingly, Applicants respectfully submit that the rejection is defective and should be withdrawn.

Further, with respect to claim 1, the Examiner correctly acknowledges the following:

> Cox fails to specifically show: the transformation rules being specific to a different graphics rendering language, whereby the transformation rules support a plurality of graphics rendering languages; and transforming the abstract data structure into a plurality of concrete data structures, each concrete data structure corresponding to a different graphics rendering language.

See Final Office Action dated December 18, 2008, page 4 (emphasis added). Thus, the Examiner proposes to modify *Cox* with *Baudel* in order to teach the limitations at issue. Specifically, the Examiner asserts as follows:

> In the same field of invention, Baudel teaches: a method for graphically rendering information of a database. Baudel further teaches: a visualization of information stored in a database (col. 1, l. 7-13), a visualization of a data table being a program that given as input any instance of a data table, outputs a uniquely defined sequence of graphic language instructions (col. 3, l. 48-50), a graphic language being a set of programming language functions and data types that enable describing images on a computer screen, examples of which OpenGL, Poscript, Java3D (col. 3, lines 22-29), and a DECORATION process setting graphic attributes for each record being drawn, said attributes including certain illumination models described in languages such as OpenGL and Java3D (i.e., because this process sets graphics attributes in languages such as OpenGL and Java3D, the visualization described inherently may be output in those languages).
>
> Thus, it would have been obvious to one of ordinary skill in the art, having the teachings of Cox and Baudel at the time that the invention was made, to have combined the visualization of information stored in a database, a visualization of a data table being a program that given as input any instance of a data table, outputs a uniquely defined sequence of graphic language instructions, a graphic language being a set of programming language functions and data types that enable describing images on a computer screen, examples of which OpenGL, Poscript, Java3D, and a DECORATION process setting graphic attributes for each record being drawn, said attributes including certain illumination models described in languages such as OpenGL and Java3D of Baudel with the method as taught by Cox.

See Final Office Action dated December 18, 2008, page 4 (emphasis added). Further, the Examiner states:

> [O]ne of ordinary skill in the art would readily equate Baudel's teachings as follows: 1) the transformation rules (e.g., the transformation from a table of data to a visualization of a table, chart or graph based on said table of data) being specific (e.g., unique) to a different graphic rendering language (e.g., OpenGL, Poscript, Java3D), whereby the transformation rules support a plurality of graphics rendering languages; 2) and transforming the abstract data structure (e.g., table of data) into a plurality of concrete data structures (e.g., visualization of a table, chart, or graph based on said table of data), each concrete data structures (e.g., visualization) corresponding to a different graphics rendering language (e.g., a DECORATION process of said visualization would have unique illumination model attribute which depends to the graphic language used; thus using a different graphic language to produce the visualization would change the visualization.

Id., page 8 (emphasis added). Applicants respectfully submit that the Examiner is mischaracterizing the limitations at issue. For example, the Examiner suggests that *Baudel* teaches *transforming the abstract data structure into a plurality of concrete data structures*. In particular, the Examiner analogizes a "table of data" in *Baudel* to teach an *abstract data structure*. In other words, the Examiner's interpretation merely equates "abstract data structure" with "data." Respectfully, the Examiner's interpretation trivializes the limitation of "generating an *abstract data structure* defining a plurality of abstract attributes representing an abstract graphical representation of the *data*" as recited in claim 1. That is, *Baudel* does not disclose any abstract data structure defining a plurality of abstract attributes representing an abstract graphical representation of a table of data. Therefore, *Baudel* does not disclose *transforming the abstract data structure into a plurality of concrete data structures*. Accordingly, Applicants respectfully submit that the rejection is defective and should be withdrawn.

Even assuming, *arguendo*, that *Baudel* somehow teaches *transforming the abstract data structure into a plurality of concrete data structures*, *Baudel* nevertheless fails to disclose *providing transformation rules for transforming the abstract data structure into a concrete data structure, the transformation rules comprising a plurality of subsets of transformation rules each subset describing graphical attributes of a requested graphical representation type and being specific to a different graphics rendering language*, as recited in claim 1. The Examiner suggests the contrary. Specifically, the Examiner states:

> Baudel explicitly teaches transformation rules (e.g., the rules used to transform a table of data into a visualization of said data), transforming the abstract data structure (e.g., table of data) into a concrete data structure (e.g., a visualization).

Final Office Action dated December 18, 2008, page 8. Curiously, the Examiner does not provide any citation for the "explicit" teaching. In fact, *Baudel* in its entirety fails to disclose *transformation rules* of any sort. In other words, the Examiner is merely inferring a teaching of "transformation rules" from "creating a visualization of a table from a table of data using graphic language instructions" in *Baudel*. However, Applicants claim "transformation rules . . . describing graphical attributes of a requested graphical representation type and being specific to a different graphics rendering language." Significantly, *Baudel* fails to disclose any transformation rules that describe graphical attributes of a requested graphical representation type or that are specific to a different graphics rendering language. Therefore, *Baudel* does not disclose *providing transformation rules for transforming the abstract data structure into a concrete data structure, the transformation rules comprising a plurality of subsets of transformation rules each subset describing graphical attributes of a requested graphical representation type and being specific to a different graphics rendering language* as recited in claim 1. Accordingly, Applicants respectfully submit that the rejection is defective and should be withdrawn.

Therefore, the claims are believed to be allowable, and allowance of the claims is respectfully requested.

Conclusion

Having addressed all issues set out in the office action, Applicants respectfully submit that the claims are in condition for allowance and respectfully request that the claims be allowed.

If the Examiner believes any issues remain that prevent this application from going to issue, the Examiner is strongly encouraged to contact Gero McClellan, attorney of record, at (336) 698-4286, to discuss strategies for moving prosecution forward toward allowance.

Respectfully submitted, and
S-signed pursuant to 37 CFR 1.4,

/Gero G. McClellan, Reg. No. 44,227/

Gero G. McClellan
Registration No. 44,227
PATTERSON & SHERIDAN, L.L.P.
3040 Post Oak Blvd. Suite 1500
Houston, TX 77056
Telephone: (713) 623-4844
Facsimile: (713) 623-4846
Attorney for Applicants